Web Interface Plug-In

This section covers the following topics:

- Before You Start
- Invoking Web Interface Plug-In Commands
- Web Interface Plug-In Functions

Before You Start

The Web Interface Plug-In is an optional plug-in unit for Natural Studio. Therefore, before any action can be taken, the Web Interface Plug-In must be activated in your Plug-In Manager. For detailed information on the activation procedure, see the section [WIN::WIN]Plug-In Manager[WIN::WIN].

Web Interface Plug-In Interface

Once Web Interface Plug-In is activated, your Natural Studio interface will be changed in the following way:

- the Tools menu will display the relevant Web Interface Plug-In commands,
- a toolbar will be available for the commands most frequently used.

The commands available depend on your working context.

Invoking Web Interface Plug-In Commands

To invoke Program Wizard commands from the main menu

1. From the main menu, choose Tools > Web Interface > Program Wizard...

The available commands are described in detail in the section Program Wizard.

To invoke DCOM Class Wizard commands from the main menu

1. From the main menu, choose Tools > Web Interface > Class Wizard...

The available commands are described in detail in the section DCOM Class Wizard.

To invoke Test Tool commands from the main menu

1. From the main menu, choose Tools > Web Interface > Test Tool...

The available commands are described in detail in the section Online Test Utility WEB-ONL.

To invoke Web Interface Plug-In commands, use the following toolbar buttons

For the Wizards:
Displays active cross-references for selected objects
Selects the HTML page that should be used for the generation process.
Starts the external editor for the selected HTML file.
Starts the import of a resource.
Starts the generation process.

For the Test Tool:

Runs the process of receiving the output from the requested subprogram. The status of the process can be seen in the status bar at the bottom of the WEB-ONL dialog window.

Starts the editor. It is disabled as long as you have not executed the program and if you have not changed the subprogram library or name. You can choose the editor within the Options dialog.

Starts the browser chosen within the Options dialog. It is disabled as long as you have not executed the program and if you have not changed the subprogram library or name. You can choose the browser in the Options dialog.

Web Interface Plug-In Functions

Program Wizard

Note:
The Program Wizard is not applicable to mainframe systems.

This section describes the use of Program Wizard, a dialog that enables you to generate basic web Interface programs and programs that use HTML templates with the Natural Web Interface.

The basic generation can be used to generate necessary parts for a subprogram called from the internet with the web interface. Then your specific coding can be added.

The template generation works with ready designed HTML pages. These HTML pages will be loaded from the resource directory and then specific parts can be replaced with your individual parts. The program generator reads these html pages, searches for the parts to be replaced (marked with special characters) and then generates an external subroutine that can be used equal to output only maps.

This section covers the following topics:

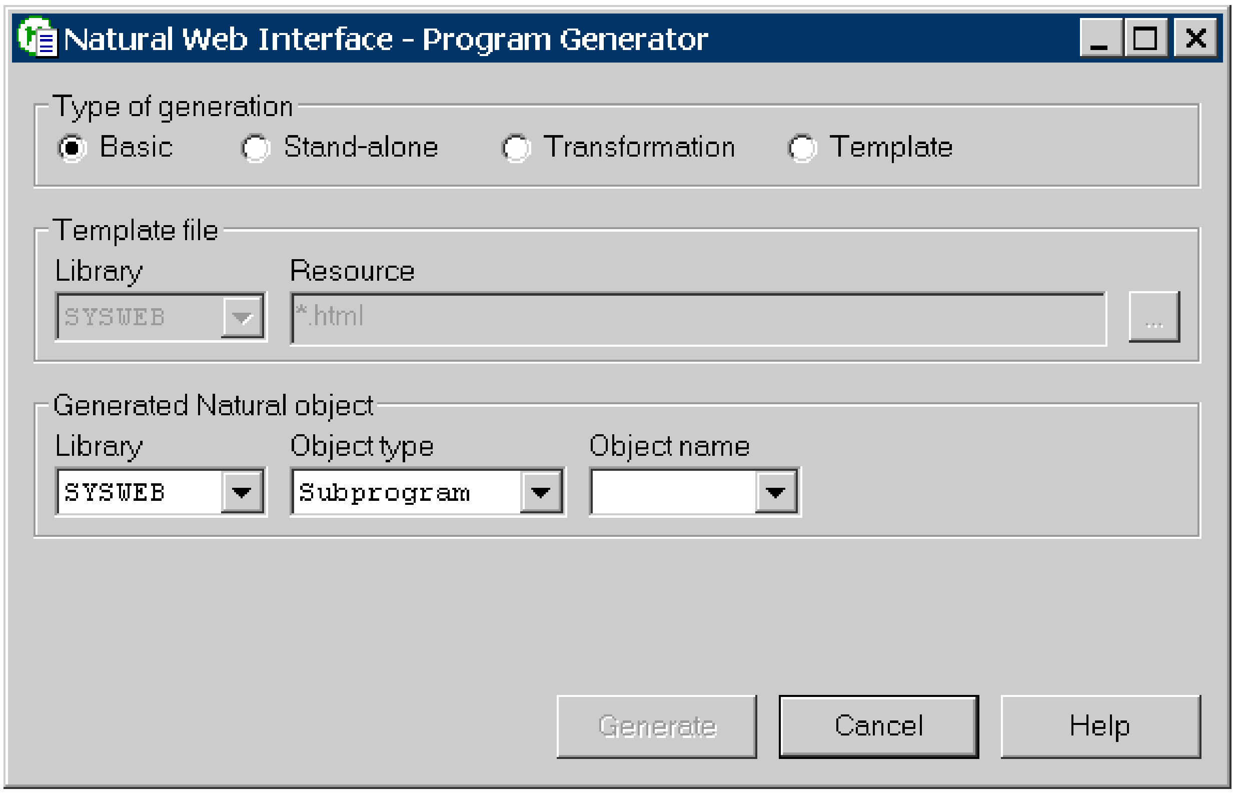
- Using the Conversion Program
- Inserting Replacement Strings
- Options
- View
- DCOM Class Wizard
- Online Test Utility WEB-ONL

Using the Conversion Program

If your basic Web pages are designed with editing tools, it takes some effort to include such a page in a Natural subprogram that can be called from the Web.

Web Wizard is a dialog that uses an HTML page as input and generates a Natural subprogram, which can be called by the Natural Web Server Extensions using the Natural Web Interface, or a subroutine which can be called to generate the output.

With the basic generation of the Web Wizard, a standalone Natural subprogram that can be called by the Natural Web Server Extensions using the Natural Web Interface can be produced.


Natural Web Interface - Program Generator

Type of generation
[✓ Basic] Stand-alone Transformation Template

Template file
Library: SYSWEB
Resource: *.html

Generated Natural object
Library: SYSWEB
Object type: Subprogram
Object name:

Generating a Basic Subprogram to be Called Direct from the Web

To generate a subprogram/subroutine to be called direct from the Web:

1. Select type of generation: basic.
2. Select your Natural file name.
3. Start the generation.
4. If you generated this subprogram the first time and you want to call the generated subprogram via DCOM, regenerate the DCOM class (see: DCOM Class Wizard).
5. After generation, this page can be called from the internet, but because this page does not set any data, the page will be empty!

Example of a basic generation.

Generated Natural subprogram, to be called directly from the internet:

```
0010 * ----- GENERATED BY NATURAL WEB INTERFACE
* Library .....: SYSPLWEB
* Source Name .: BASIC
* -----
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
* ----- PRIVATE VARIABLES -----
* LOCAL
* 1 W3VALUE        (A250)
END-DEFINE
* ----- ERROR HANDLER -----
ON ERROR
PERFORM W3ERROR ##W3ERROR
PERFORM W3END ##RPC
ESCAPE ROUTINE
END-ERROR
* ----- INITIALISE HTTP API -----
PERFORM W3INIT ##RPC
* --- READ ENVIRONMENT ---
* PERFORM W3READ-ENVIRONMENT-DYNAMIC 'varname' ' ' W3VALUE
* set default value
* IF *length(W3VALUE) = 0 THEN
*  W3VALUE := ??
* END-IF
* ----- HEADER FOR SERVER -----
* PERFORM W3CONTENT-TYPE 'text/html'
*
*
* Add your individual coding using W3* subroutines or
* call your own subroutines.
*
*
* ----- END HTTP -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
END
```

Generating a Standalone Subprogram to be Called Direct from the Web

To generate a subprogram to be called direct from the Web:

1. Select type of generation: standalone.
2. Select your Natural file name.
3. Start the generation.
4. If you generated this subprogram the first time and you want to call the generated subprogram via DCOM, regenerate the DCOM class (see: DCOM Class Wizard).
5. After generation, you can call the Natural Web Interface to show the page.

Example of a standalone generation.

Generated Natural subprogram, to be called directly from the internet:

```
* ----- GENERATED BY NATURAL WEB INTERFACE
* Library .....: SYSPLWEB
* Source Name .: ALONE
* -----
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
* ----- PRIVATE VARIABLES -----
LOCAL
1 W3VALUE        (A250)
END-DEFINE
* ----- ERROR HANDLER -----
ON ERROR
PERFORM W3ERROR ##W3ERROR
PERFORM W3END ##RPC
 ESCAPE ROUTINE
END-ERROR
* ----- INITIALISE HTTP API -----
PERFORM W3INIT ##RPC
* ----- HEADER FOR SERVER -----
PERFORM W3CONTENT-TYPE 'text/html'
*
* --- READ ENVIRONMENT ---
* PERFORM W3READ-ENVIRONMENT-DYNAMIC 'varname' ' ' W3VALUE
*  set default value
* IF *length(W3VALUE) = 0 THEN
*  W3VALUE := ??
* END-IF
* --- WRITE THE HEAD OF THE DOCUMENT ---
PERFORM W3TEXT "<!DOCTYPE 'HTML PUBLIC-//W3C//DTD HTML 3.2//EN'>"-
'<html>'-
'<head>'-
"<meta http-equiv='Content-Type' content='"-
"text/html; charset=iso-8859-1'>"-
'<title>SYSPLWEB/TEST</title>'-
'</head>'
* --- WRITE THE BODY OF THE DOCUMENT ---
PERFORM W3TEXT '<body>'-
'<h2>SYSPLWEB/TEST</h2>'-
'<hr>'
*
PERFORM W3TEXT '<p>This is your output</p>'
*
COMPRESS '<hr>generated:' *DATE *TIME INTO W3VALUE
PERFORM W3TEXT W3VALUE
* --- END THE BODY OF THE DOCUMENT ---
PERFORM W3TEXT '</body>'-
'</html>'
*
* ----- END HTTP -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
END
```

Generating a Subprogram/Subroutine using Natural Tags

To generate a subprogram/subroutine to be called direct from the Web:

1. Select type of generation: tranformation.
2. Select your HTML page.
3. You can view your selected HTML page with an editor/browser.
4. Close the Natural library you want to generate to.
5. Select the object type you want to generate.
6. Select your Natural file name.
7. Start the generation.
8. If you generated this subprogram the first time and you want to call the generated subprogram via DCOM, regenerate the DCOM class (see: DCOM Class Wizard).
9. After generation, you can call the Natural Web Interface to show the page.

Inserting a Natural Tag

It is possible to specify Natural coding directly in the HTML page. After generation, the program needs no additional changes.

The HTML2NAT dialog can recognize a <NATURAL> tag. All lines between <NATURAL> and </NATURAL> will be copied, as they are, to the generated Natural source object.

Appearance

```
<NATURAL> </NATURAL>
```

Below is information on:

- Attributes DATA, LDA, GDA, SUB, NOT
- Comment Tag
- ASP-like Script Commands
- Additional Script Directives
- Example of a Simple Generation
- Example of a Simple Generation with a Natural Tag

Attributes DATA, LDA, GDA, SUB, NOT

Listed below are attributes provided to define coding sections that are to be moved within the program or excluded from the program.

Attribute	Explanation
DATA	<NATURAL DATA> or <NATURAL LDA> moves the defined section to the DEFINE DATA LOCAL part of your program.
LDA	
GDA	<NATURAL GDA> moves the defined section to the DEFINE DATA GLOBAL part of your program.
SUB	<NATURAL SUB> moves the defined section to the end of the program. This enables you to specify inline subroutines.
NOT	<NATURAL NOT> excludes the defined section from the program. This enables you to specify the design of part of a page that will be generated by a program.

Comment Tag

Use the comment tag "<!-- -->" to hide the display of defined sections of your coding. If you use the comment tag and <NATURAL NOT>, you can display the predefined page with a normal browser. This helps you to specify your page and replace parts of the page dynamically.

ASP-like Script Commands

Not only <NATURAL> and </NATURAL> can be used, but also ASP-like (Active Server Pages) script commands which are differentiated from the text by using the "<%" and "%>" delimiters.

Additional Script Directives

The following Natural-specific directives must be used when writing a Natural subprogram:

Output directive: <%= ... %>
    Short form for <% PERFORM W3HTML ... %> tag
Subprogram directive: <%SUB ... %>
    equal to the <NATURAL SUB> ... </NATURAL> tag
Global Data Area directive: <%GDA ... %>
    equal to the <NATURAL GDA> ... </NATURAL> tag
directive: <%LDA ... %>
    equal to the <NATURAL LDA> ... </NATURAL> tag
Not directive: <%NOT ... %>
    equal to the <NATURAL NOT> ... </NATURAL> tag
Processing directive <%@ LANGUAGE=NATURAL %>
    indicates that the used language is Natural.

Example 1 of a Simple Generation

HTML document:

```
 <HTML><HEAD><TITLE>
 Example1 genNat
</TITLE></HEAD><BODY><H2>
 Example1 genNat
 </H2><HR>
 <P>This is for your output
 </BODY></HTML>
```

Generated Natural subprogram:

```
* ----- GENERATED BY NATURAL WEB INTERFACE
* File ..........: E:\SAG\Natural\6.1.1\Fnat\SYSWEB\RES\example1.html
* Library .......: SYSWEB
* Source Name ...: EXAMPLE1
* Crunch Lines...: 1
* Save Source....: 1
* Line Length....: 128
* Long Constants.: 1
* -----
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
LOCAL
* ----- PRIVATE VARIABLES -----
1 W3VALUE (A250)
END-DEFINE
*
* ----- ERROR HANDLER -----
ON ERROR
  PERFORM W3ERROR ##W3ERROR
  PERFORM W3END ##RPC
  ESCAPE ROUTINE
END-ERROR
* ----- INITIALIZE HTTP API -----
PERFORM W3INIT ##RPC
* ----- HEADER FOR SERVER -----
PERFORM W3CONTENT-TYPE ’text/html’
*
* ----- MAIN PROGRAM -----
PERFORM W3TEXTLINE’<HTML><HEAD><TITLE>’
PERFORM W3TEXTLINE ’Example genNat’
PERFORM W3TEXTLINE’</TITLE></HEAD><BODY><H2>’
PERFORM W3TEXTLINE ’Example genNat’
PERFORM W3TEXTLINE ’</H2><HR>’
PERFORM W3TEXTLINE ’<P>This is for your output’
PERFORM W3TEXTLINE ’</BODY></HTML> ’
* ----- END HTTP API -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
*
* ----- SUBROUTINES -----
END
```

Example 2 of a Simple Generation with a Natural Tag

HTML document:

```
 <HTML><HEAD><TITLE>
 Example2 genNat
</TITLE></HEAD><BODY><H2>
 Example2 genNat
 </H2><HR>
 <P>This is for your output
 <HR>
 <P>generated at:
 <NATURAL NOT>
 Time/Date
 </NATURAL>
 <NATURAL><!--
  PERFORM DOTIME
 --></NATURAL>
 <NATURAL SUB><!--
 DEFINE SUBROUTINE DOTIME
  COMPRESS *TIME *DATE INTO #VALUE
  PERFORM W3TEXTLINE #VALUE
 END-SUBROUTINE
 --></NATURAL>
 <NATURAL DATA><!--
 1 #VALUE (A30)
 --></NATURAL>
 </BODY></HTML>
```

Generated Natural subprogram:

```
* ----- GENERATED BY NATURAL WEB INTERFACE
* File ..........: E:\SAG\Natural\6.1.1\Fnat\SYSWEB\RES\example2.html
* Library .......: SYSWEB
* Source Name ...: EXAMPLE2
* Crunch Lines...: 1
* Save Source....: 1
* Line Length....: 128
* Long Constants.: 1
* -----
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
1 #VALUE (A30)
* ----- PRIVATE VARIABLES -----
1 W3VALUE (A250)
END-DEFINE
*
* ----- ERROR HANDLER -----
ON ERROR
  PERFORM W3ERROR ##W3ERROR
  PERFORM W3END ##RPC
  ESCAPE ROUTINE
END-ERROR
* ----- INITIALIZE HTTP API -----
PERFORM W3INIT ##RPC
* ----- HEADER FOR SERVER -----
PERFORM W3CONTENT-TYPE ’text/html’
*
* ----- MAIN PROGRAM -----
PERFORM W3TEXTLINE’<HTML><HEAD><TITLE>’
PERFORM W3TEXTLINE ’Example2 genNat’
PERFORM W3TEXTLINE’</TITLE></HEAD><BODY><H2>’
PERFORM W3TEXTLINE ’Example2 genNat’
PERFORM W3TEXTLINE ’</H2><HR>’
PERFORM W3TEXTLINE ’<P>This is for your output’
PERFORM W3TEXTLINE ’<HR>’
PERFORM W3TEXTLINE ’<P>generated at:’
 PERFORM DOTIME
PERFORM W3TEXTLINE ’</BODY></HTML>’
* ----- END HTTP API -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
*
* ----- SUBROUTINES -----
DEFINE SUBROUTINE DOTIME
 COMPRESS *TIME *DATE INTO #VALUE
 PERFORM W3TEXTLINE #VALUE
END-SUBROUTINE
END
```

Note:
The syntax of the Natural program will not be checked during conversion.

To Generate a Subprogram/Subroutine using a Template that is Called Direct from the Web

To generate a subprogram/subroutine using a template that is called direct from the Web:

1. Select type of generation: template.
2. Select your HTML page.
3. You can view your selected HTML page with an editor/browser.
4. Select the object type you want to generate.
5. Select your Natural file name.
6. Start the generation.
7. If you generated this subprogram the first time and you want to call the generated subprogram via DCOM, regenerate the DCOM class (see: DCOM Class Wizard).
8. After generation, you can call the Natural Web Interface to show the page.

Inserting Replacement Strings

It is neccesary to specify the replacement strings directly in the HTML page. The replacement strings have to start and end with an specific character, e.g. $ (see options). The name (content) of a string has to comply with the Natural rules for variable names. If not, subroutines may not stow.

If the name of the replacement string is prefixed with "HTML", unsaved characters as "<" or ">" will be replaced during replacement at runtime.

The following prefixes for automatic conversion at runtime are implemented:

- HTML
- URL
- XML

For more information, see the documentation of the subroutine W3REPLACE-AT-OUTPUT.

Example of Template Generation

HTML document:

```
<HTML>
<HEAD>
  <TITLE>Template Processing</TITLE>
</HEAD>
<BODY>
<H2>
  Template Processing
</H2>
<P>
  <HR>
<TABLE BORDER="0">
<TR><TD>Log-Time:</TD><TD>$log$<TD></TR>
<TR><TD>HTTPs Extension:</TD><TD>$html-ext$<TD></TR>
<TR><TD>Web Interface:</TD><TD>$html-ver$<TD></TR>
</TABLE>
<P>
<TABLE BORDER='0' WIDTH='100%' CELLSPACING='0' CELLPADDING=5>
  <TR BGCOLOR='#00cc66'>
    <TD>$prog$ - $log$</TD>
    <TD ALIGN='RIGHT'>Natural</TD>
  </TR>
</TABLE>
</BODY></HTML>
```

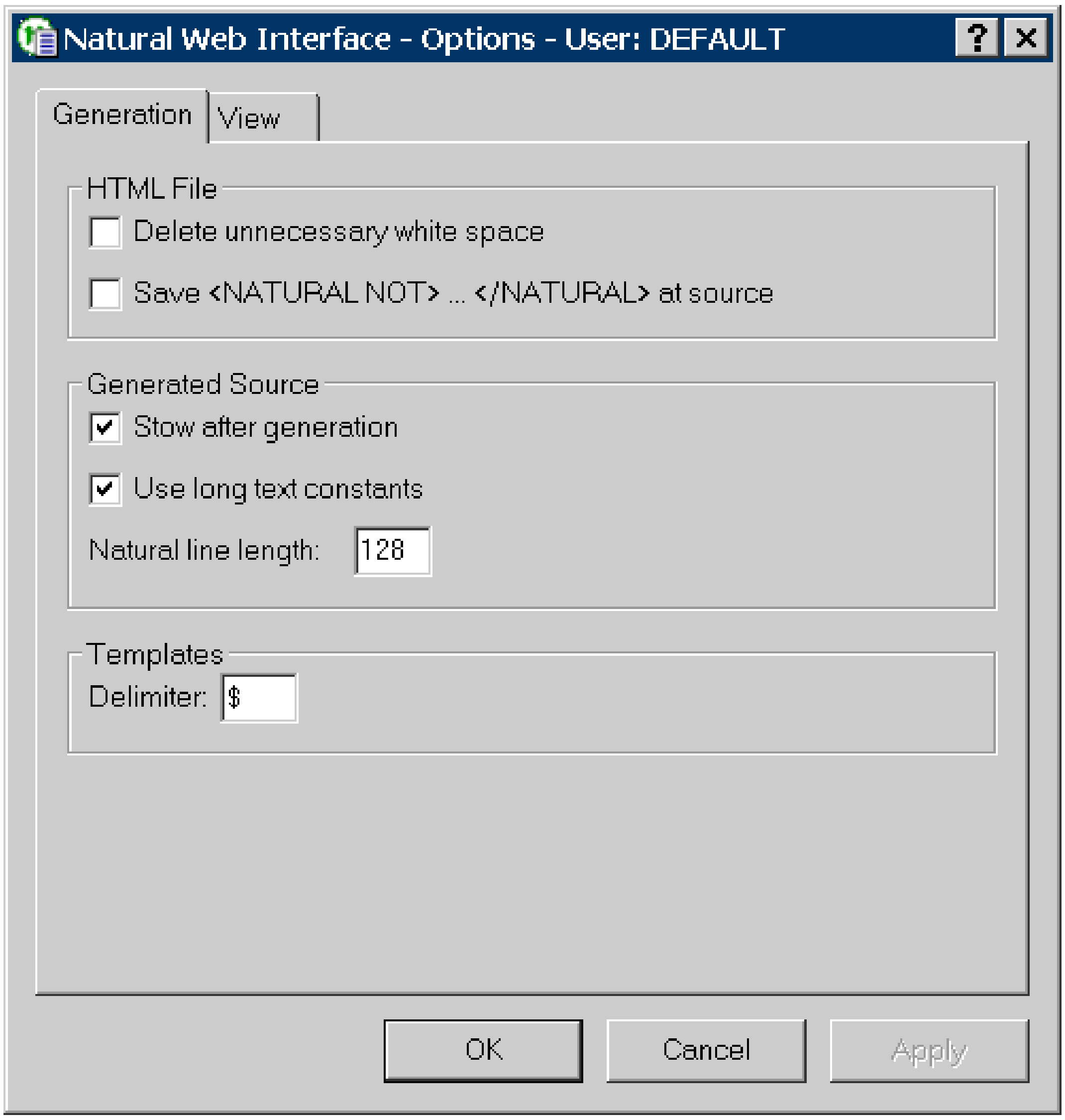
Generated Natural subroutine, that has to be called from a subprogram that is called from the internet:

```
 * ----- GENERATED BY NATURAL WEB INTERFACE
* File ........: E:\SAG\Natural\6.1.1\Fnat\SYSWEB\RES\templ.html
* Library .....: SYSWEB
* Source Name .: TEMPL
* Delimiter ...: $
* -----
DEFINE DATA PARAMETER
1 log                                (A) DYNAMIC BY VALUE
1 html-ext                           (A) DYNAMIC BY VALUE
1 html-ver                           (A) DYNAMIC BY VALUE
1 prog                               (A) DYNAMIC BY VALUE
END-DEFINE
*
*
DEFINE SUBROUTINE e3templm
*
* ----- HEADER FOR SERVER -----
PERFORM W3CLEAR
PERFORM W3CONTENT-TYPE 'text/html'
* ----- MAIN PROGRAM -----
* --- LOAD THE HTML TEMPLATE ---
PERFORM W3LOAD-RESOURCE ' '   'e3templ.html'
*
* --- REPLACE PLACEHOLDER ---
PERFORM W3REPLACE-AT-OUTPUT ' ' '$log$' log
PERFORM W3REPLACE-AT-OUTPUT 'HTML' '$ext$' ext
PERFORM W3REPLACE-AT-OUTPUT 'HTML' '$ver$' ver
PERFORM W3REPLACE-AT-OUTPUT ' ' '$prog$' prog
* ----- END MAIN PROGRAM -----
*
END-SUBROUTINE
*
END
```

Generated Natural subprogram, to be called directly from the internet:

```
* ----- GENERATED BY NATURAL WEB INTERFACE
* File ........: E:\SAG\Natural\6.1.1\Fnat\SYSWEB\RES\templ.html
* Library .....: SYSWEB
* Source Name .: TEMPL
* Delimiter ...: $
* -----
DEFINE DATA
PARAMETER USING W3PARM
LOCAL USING W3CONST
LOCAL
* ----- PRIVATE VARIABLES -----
1 W3VALUE        (A250)
END-DEFINE
*
* ----- ERROR HANDLER -----
ON ERROR
 PERFORM W3ERROR ##W3ERROR
 PERFORM W3END ##RPC
 ESCAPE ROUTINE
END-ERROR
* ----- INITIALISE HTTP API -----
PERFORM W3INIT ##RPC
* ----- HEADER FOR SERVER -----
PERFORM W3CONTENT-TYPE ’text/html’
*
* ----- MAIN PROGRAM -----
* --- LOAD THE HTML TEMPLATE ---
PERFORM W3LOAD-RESOURCE ’SYSWEB’   ’e3templ.html’
*
* --- REPLACE PLACEHOLDER ---
PERFORM W3REPLACE-AT-OUTPUT ’ ’ ’$log$’ ’replace-string-1’
PERFORM W3REPLACE-AT-OUTPUT ’HTML’ ’$ext$’ ’replace-string-2’
PERFORM W3REPLACE-AT-OUTPUT ’HTML’ ’$ver$’ ’replace-string-3’
PERFORM W3REPLACE-AT-OUTPUT ’ ’ ’$prog$’ ’replace-string-4’
* ----- END HTTP -----
PERFORM W3END ##RPC
* ----- END MAIN PROGRAM -----
*
END
```

Options


Natural Web Interface - Options - User: DEFAULT

HTML File

- [ ] Delete unnecessary white space
- [ ] Save <NATURAL NOT> ... </NATURAL> at source

Generated Source

- [x] Stow after generation
- [x] Use long text constants

Natural line length: 128

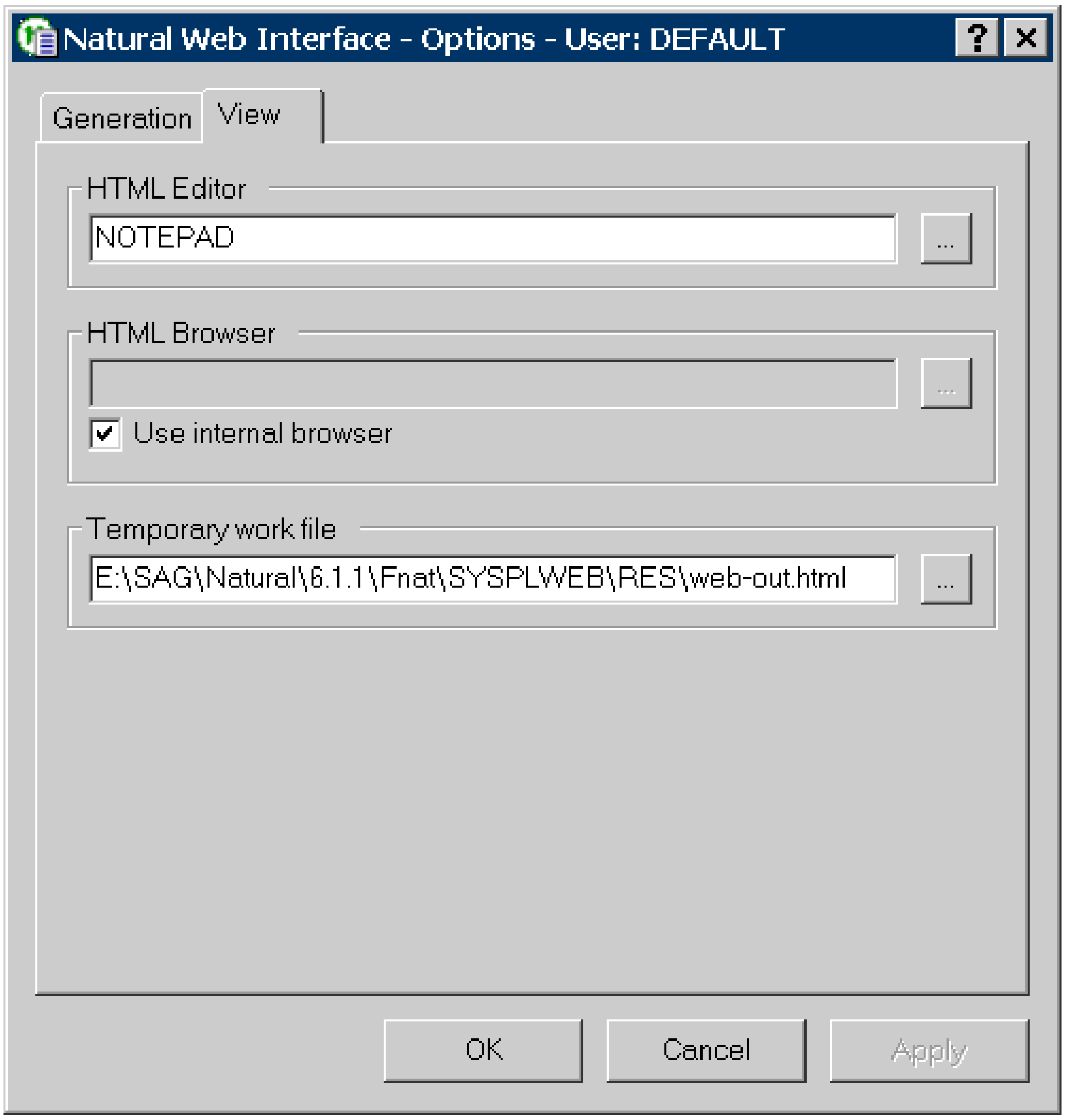
Templates

Delimiter: $

Input/Output Fields

Field	**Explanation**
Delete unnecessary white space	If checked, multiple white-space characters such as blank, new line, tab, will be reduced to a single white space. For special HTML tags such as <PRE> <TEXTAREA> or <SCRIPT>, the white space will not be collapsed. Default value: unchecked
Save <NATURAL NOT> ... <NATURAL> in Source	If checked, the content of <NATURAL NOT> tags will not usually be generated into the Natural source. This option generates the content of <NATURAL NOT> as comment into the Natural source. Default value: unchecked
Stow after Generation	If checked, the generated program will be stowed if the generation has been successful. Default value: checked
Use Long Text Constants	Generate text constants longer than 253 characters for better performance. Default value: checked
Natural Line Length	The length of the generated Natural source lines: the minimum value is 20, the maximum 248. Default value: platform dependant
Delimiter	Delimiter string for replacement strings. Default value: $

View


Natural Web Interface - Options - User: DEFAULT

HTML Editor: NOTEPAD

HTML Browser:

[✓] Use internal browser

Temporary work file: E:\SAG\Natural\6.1.1\Fnat\SYSPLWEB\RES\web-out.html

Below is information on:

- Input/Output Fields
- Buttons

Input/Output Fields

Field	**Explanation**
HTML Editor	The external program that is used to edit the source of the HTML page. Default Value: NOTEPAD
HTML Browser	The external program that is used to display the HTML page. Default: Microsoft Web browser ActiveX Control
Use internal Browser	The external program that is used to display the HTML page. To select your own browser, uncheck this box. Default value: checked
Web Interface Server	The URL to the HTTP server when installed. If the Natural Web Interface has been configured correctly, the name of the library and the subprogram are added to this URL. The browser will then be executed to test the generated program. Default value: http://localhost/cgi-bin/nwwdcgi.exe
Default Input File	The default input file to be used for the generation. Default value: C:\Program Files ...*.html
Temporary Workfile	The default output file to be used for displaying data in the HTML-browser. Default value: C:\Temp\web-out

Buttons

OK	Leaves the dialog and saves the changes.
Cancel	Leaves the dialog without saving your changes.
Apply	Saves the current input.

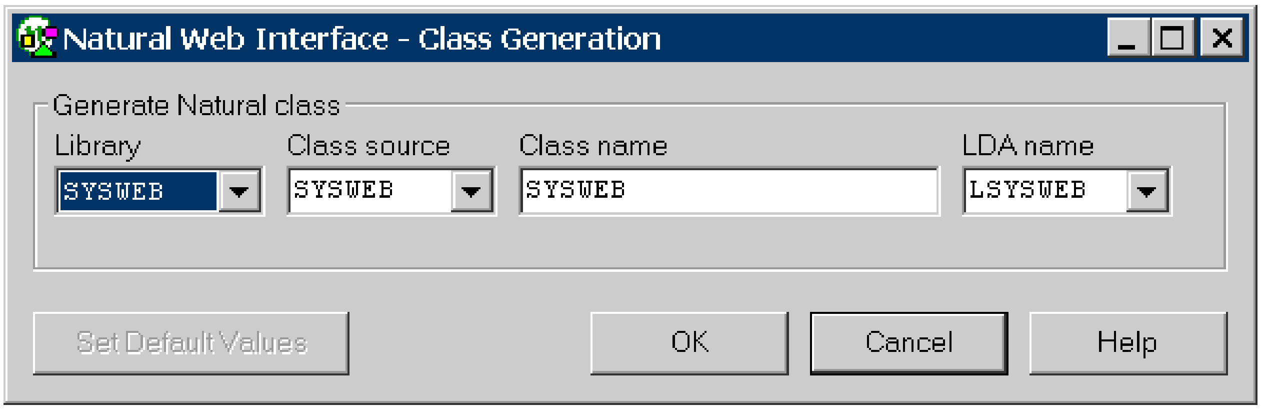
DCOM Class Wizard

If the Natural Web Interface subprograms should be called using DCOM instead of RPC, a DCOM class is needed. This class contains as methods all relevant Natural subprograms for the Natural Web Interface.

The program HTML to Natural automatically generates the specified class. To stow the generated class, a Local Data Area (LDA) is needed to specify the Global Unique IDs (GUIDs) of the DCOM objects. The name of the LDA begins with L followed by the first seven characters of Library Name.

Below is information on:

- Input/Output Fields
- Buttons



Input/Output Fields

Field	Explanation
Library Name	The name of the library to be scanned.
Class Source	The name of the class source. We recommend that the name you choose for Class Source is identical to the name of the library.
Class Name	The name of the class that can be called later from the Internet. We recommend that the name you choose for Class Name is identical to the name of the library for which the class is generated.
LDA Name	The name of the LDA containing the GUIDs for the class ID and the Natural Web Interface ID. For the naming conventions that apply, see Example for Library SYSWEB below.

Buttons

OK	Generates the class and leaves the dialog.
Cancel	Leaves the dialog without generation.
Set Default Values	This button is enabled if no relevant class is found for the library. The defaults for class source and class name are given. The LDA needed has to be generated in advance.

Example for Library SYSWEB

The LDA name is LSYSWEB. Name the first GUID CLSID- followed by the Library name and the second GUID IID-NATWEB.

	T		Comment			
			*** Top of Data Area ***			
X	U	1	CLSID-**SYSWEB**	A	36	
X	U	1	IID-NATWEB	A	36	
			*** End of Data Area ***			

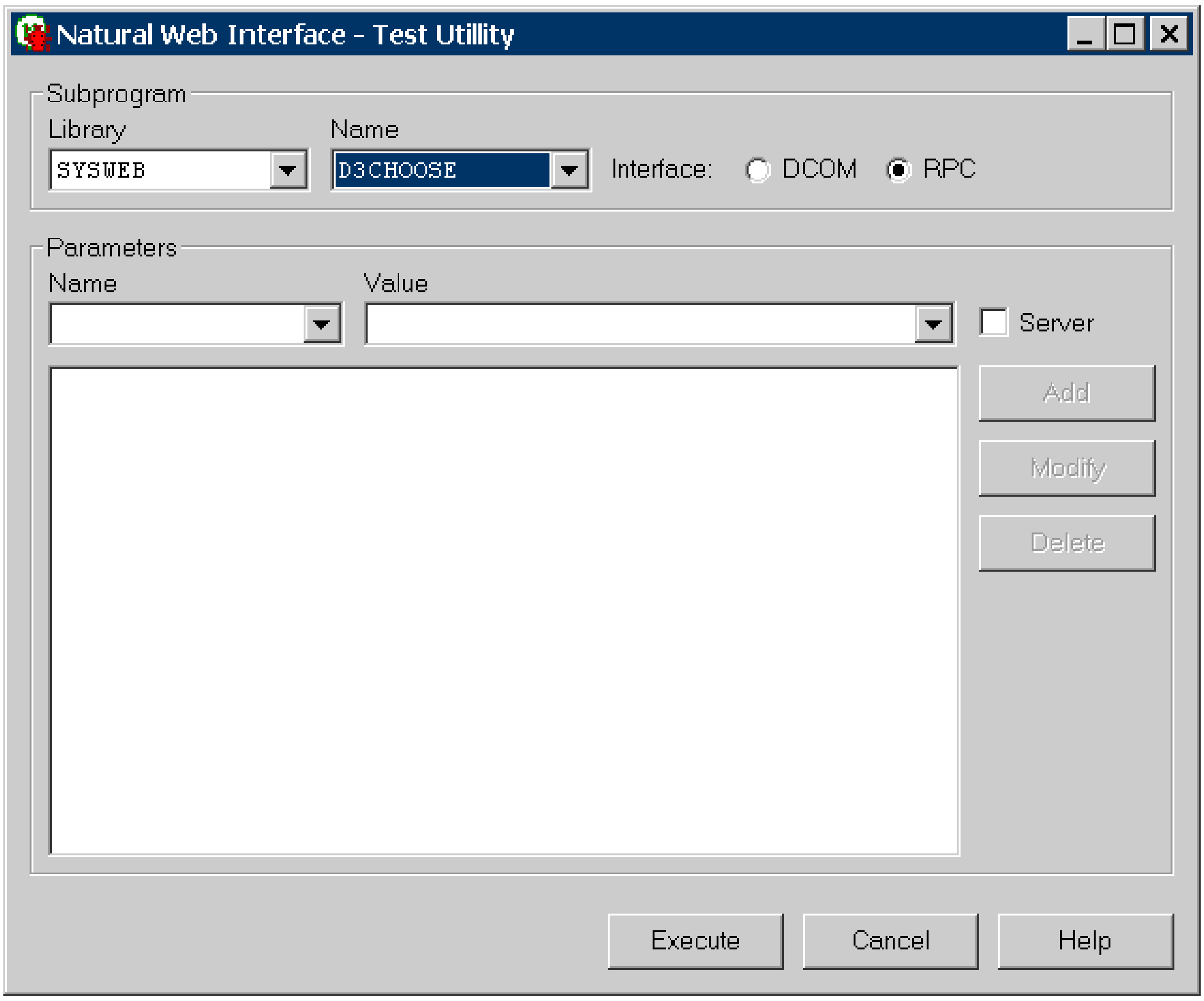
Attention:
Do **not** copy and rename or move an LDA in order to get new GUIDs for your classes. If an LDA is copied and renamed or moved, the preset GUID is not changed. This may cause major problems.

Online Test Utility WEB-ONL

This Test Utility is a component of Natural Web Interface. You have the ability to check your subprogram locally without involving an HTTP server. The transfer parameters for your Web page are transferred into the Test Utility and are posted directly to the business logic. As communication platform, you can choose either RPC or DCOM as in real remote communications. The result is either the Web page expected or an error message. The Web page can be viewed with the browser or a viewer of your choice. If you receive an error message, you can easily debug your business logic locally without writing an extra test routine. No remote debugging is needed.

Features:

- Local application checking.
- No need for remote debugging.
- Simplified error checking.
- Comfortable operation by user friendly interface.
- No need to write an extra test routine.



Library: SYSWEB
Name: D3CHOOSE
Interface: DCOM [✓ RPC]

Below is information on:

- Prerequisites
- Running the Application
- Supported Content Types
- Input/Output Fields
- Buttons

Prerequisites

- Web browser which supports different content types, for example, Microsoft Internet Explorer Version 5.0 or higher.
- Any available text editor.

Running the Application

To define path adjustments

1. Start the main dialog.
2. Select a browser and viewer of your choice with Tools > Options...
3. Set the browser, viewer and work file path.
4. Press the OK button.

To start the application

1. Start the dialog WEB-ONL.
2. Select a library and subprogram name.
3. Optional: add parameters.
4. Choose RPC or DCOM.
5. Press the Execute button.
6. View the result by pressing either the Result... or the Browse button.

Supported Content Types

The following Content Types are supported by the Test Utility:

Content Type	Extension
"application/rtf"	"rtf"
"application/powerpoint"	"ppt"
"application/msword"	"doc"
"application/excel"	"xls"
"text/html"	"htm"
"text/plain"	"txt"
"text/xml"	"xml"
"text/richtext"	"rtf"

If you need further Content Types, change the subroutine HTML2CONTENT-TYPE (SYSWEB/W3CO2EXT) and extend the translation table to suit your own needs.

Input/Output Fields

Field	Explanation
Subprogram: Library Name	Enables you to specify the library and the name of the required subprogram. The available libraries and subprograms are automatically taken from the library workspace and listed in selection boxes.
Server	If any of the name-value-pairs are server variables, you need to check this toggle button. Note that any status will last until you change it again.
Call with	Can be selected with either DCOM or RPC as communication form. For DCOM, you have to register your classes first. Default: RPC
Parameter: Name Value	Here you can enter the name-value-pairs needed from the subprogram. To take them over into the parameter list, press the Add button. To modify the entries, use the Modify button. You do not have to substitute &, =, %; this will be done by the WEB-ONL program. If you use server parameters, check the Server toggle button before you add the parameter to the parameter list. In the parameter list, all name-value-pairs are displayed. &, =, % are substituted. To delete a pair, select the item and press the Delete button. Every selected item will be inserted into the Name and Value fields. If you wish to modify a pair, select the item, change it in the Name and Value fields and press the Modify button.

Buttons:

	Runs the process of receiving the output from the requested subprogram. The status of the process can be seen in the status bar at the bottom of the WEB-ONL dialog window.
	Starts the editor. It is disabled as long as you have not executed the program and if you have not changed the subprogram library or name. You can choose the editor with the Options dialog.
	Starts the browser chosen with the Options dialog. It is disabled as long as you have not executed the program.